

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2002-318818

(43)Date of publication of application : 31.10.2002

(51)Int.Cl.

G06F 17/30
G06F 17/16
G06T 7/00

(21)Application number : 2001-122755

(71)Applicant : CANON INC

(22)Date of filing : 20.04.2001

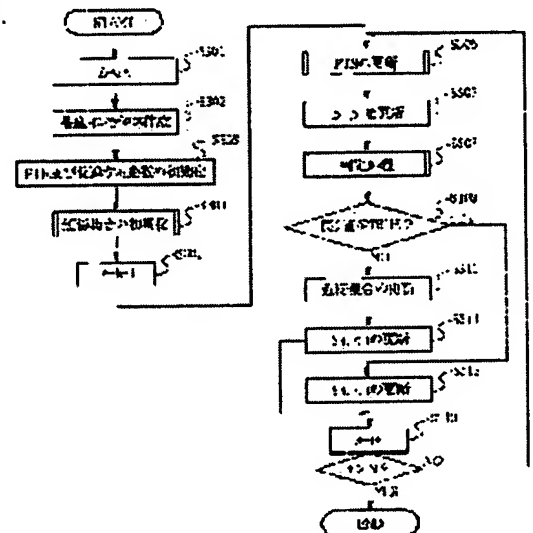
(72)Inventor : WASHISAWA TERUYOSHI

(54) DATA PROCESSING DEVICE AND ITS METHOD, AND ITS PROGRAM

(57)Abstract:

PROBLEM TO BE SOLVED: To make data retrieval executable at a high speed by similarity based on an inner product relative to given vector data.

SOLUTION: A sorting list relative to each component of a vector which is to be a prototype is made, and the nearest value to m-component values of test vectors X and -X are retrieved from the list, and respective positions are housed in PTR+ and PTR- (S303). A neighboring set to the number of k is initialized (S304), and a variable related to PTR is renewed (S306). Prototypes Yt are taken out successively, and determination processing is executed. If the similarity is larger than the minimum value up to the present, the prototype corresponding to the minimum value is replaced with Yt, and the neighboring set and the minimum value are renewed (S308-311).



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

* NOTICES *

Japan Patent Office is not responsible for any damages caused by the use of this translation.

1. This document has been translated by computer. So the translation may not reflect the original precisely.
2. **** shows the word which can not be translated.
3. In the drawings, any words are not translated.

CLAIMS

[Claim(s)]

[Claim 1] It is the data processor which extracts a prototype of a predetermined individual similar to the 1st given test vector from a prototype set described by multi-dimension vector as a retrieval result. A pretreatment means to create a list which arranged an identifier and a component value of each prototype of said prototype set in order of a component value about each component of said multi-dimension vector, and to calculate a square of a norm of each prototype, A test vector generation means to generate the 2nd test vector from said 1st test vector, A similarity count means to calculate similarity based on an absolute value of an inner product of the 1st test vector and each prototype concerned using a square of said list and a norm of each of said prototype, and the 1st and 2nd test vectors, A data processor characterized by having an abbreviation control means which omits future processings, and an updating means to update a retrieval result based on a value of said similarity, by decision of abbreviation conditions based on a value of said similarity.

[Claim 2] A data processor according to claim 1 characterized by the above-mentioned test vector generation means generating said 1st test vector and a vector which a norm is equal and is reverse as said 2nd test vector.

[Claim 3] When a dimension of a vector space as which said prototype and 1st test vector are expressed is set to d, said similarity count means If similarity is first calculated to 1-dimensional subspace, said abbreviation control means judges that of abbreviation conditions based on similarity in the 1-dimensional subspace concerned and these abbreviation conditions are not satisfied It is the data processor according to claim 1 which controls said similarity count means to raise a dimension of subspace and to perform similarity count, and is characterized by said updating means updating a retrieval result when not satisfying abbreviation conditions in addition, even if a dimension of subspace reaches d.

[Claim 4] A data processor according to claim 1 with which said similarity count means is characterized by calculating similarity using the 2nd test vector and a thing which reduced a square of a difference of a prototype from a square of the 1st test vector, a thing which reduced a square of a difference of a prototype, and a norm of a prototype from a square of a norm of a prototype.

[Claim 5] A data processor according to claim 1 characterized by having a recognition means to recognize a class to which said 1st test vector belongs, based on a class to which a prototype of said predetermined individual extracted as a retrieval result belongs.

[Claim 6] It is the data-processing method of extracting a prototype of a predetermined individual similar to the 1st given test vector from a prototype set described by multi-dimension vector as a retrieval result. A head end process which creates a list which arranged an identifier and a component value of each prototype of said prototype set in order of a component value about each component of said multi-dimension vector, and calculates a square of a norm of each prototype, A test vector generation production process which generates the 2nd test vector from said 1st test vector, A similarity count production process which calculates similarity based on an absolute value of an inner product of the 1st test vector and each prototype concerned using a square of said list and a norm of each of said prototype, and the 1st and 2nd test vectors, A data-processing method characterized by having an abbreviation control production process of omitting future processings, and an updating production process which updates a retrieval result based on a value of said similarity by decision of abbreviation conditions based on a value of said similarity.

[Claim 7] It is the data-processing program which extracts a prototype of a predetermined individual similar to the 1st given test vector from a prototype set described by multi-dimension vector as a retrieval result. A head end process which creates a list which arranged an identifier and a component value of each prototype of said prototype set in order of a component value about each component of said multi-dimension vector, and calculates a square of a norm of each prototype, A test vector generation production process which generates the 2nd test vector from said 1st test vector, A similarity count production process which calculates similarity based on an absolute value of an inner product of the 1st

test vector and each prototype concerned using a square of said list and a norm of each of said prototype, and the 1st and 2nd test vectors, A data-processing program for making a computer perform an abbreviation control production process of omitting future processings, and an updating production process which updates a retrieval result based on a value of said similarity by decision of abbreviation conditions based on a value of said similarity.

[Translation done.]

* NOTICES *

Japan Patent Office is not responsible for any damages caused by the use of this translation.

1. This document has been translated by computer. So the translation may not reflect the original precisely.
2. **** shows the word which can not be translated.
3. In the drawings, any words are not translated.

DETAILED DESCRIPTION

[Detailed Description of the Invention]

[0001]

[The technical field to which invention belongs] This invention relates to the equipment which searches or recognizes data by the similarity based on the inner product between the data expressed especially by the vector about the data processor which performs retrieval or recognition of data.

[0002]

[Description of the Prior Art] The inner product is widely used as similarity between the data expressed by the vector. For example, in character recognition or a voice recognition system, the sampled data is mapped to the characteristic quantity space stretched at the suitable base, and the data by which vectorial representation was carried out is memorized as a prototype. The inner product of the data and the prototype which were newly inputted is calculated, and input data is identified as a thing belonging to the class corresponding to the nearest prototype. Most, the bad count method of effectiveness is exhaustive search, and the computational complexity becomes the order of the dimension of a vector, and the product of the number of prototypes.

[0003] it comes out in data base retrieval that the computational complexity of an inner product is recognized as a decisive failure. A data base can store now not only document data but an image and non-text data called voice by fast development of computer processing capacity in recent years. A keyword must be given in order to search non-[these] text data with the conventional keyword. If you want to avoid the time and effort of keyword grant, similar retrieval by the characteristic quantity vector must be performed.

[0004] Moreover, even when searching document data, in order to enable more flexible retrieval, a document is expressed by the vector, and the algorithm which performs similar retrieval based on this vector is also realized. At this time, said computational complexity poses an essential problem of retrieval system implementation. The data number of cases stored in the usual data base exceeds hundreds of thousands of affairs. Therefore, it becomes the hopeless condition that computational complexity increases hundreds of thousands of times whenever one number of dimension of a vector increases. In order to avoid such a condition, the number of data which reduces the dimension of a vector or is calculated must be reduced.

[0005] Since the former is equal to reducing the dimension of the presentation space of data, information required for data retrieval may not be enough expressed as a component of a vector. The latter is methodology which as for the number of data demanded as a retrieval result has a meaning when sufficiently small compared with the total of data. The problem dealt with by k-NN retrieval is this kind of thing, and some effective methods are proposed.

[0006] k-NN retrieval is a method for searching for k near a test vector and identifying the class of a test vector based on those classes from the set of the prototype memorized by the system. In such a case, it is one of the important technical problems how k prototypes near a test vector can be found at a high speed. Such a demand exists also in data base retrieval. A retrieval user wants only some data near the search key which he specified among a lot of data stored in the data base, and no value is found out to other data, furthermore the value of an inner product, etc. The technology for meeting such a demand of a retrieval user is in agreement with the purpose of the high-speed algorithm of k-NN retrieval.

[0007] When extracting k near a test vector from the set of a prototype, in order to mitigate the computational complexity of retrieval, it is common to structure the set of a prototype beforehand. Mitigation of the computational complexity of retrieval is expectable, so that it will carry out, if the property of data is made to reflect in the case of structuring. For example, in the case where a prototype is structured hierarchical, it is attained by repeating recursively actuation of dividing N dimension vector space where the prototype is expressed.

[0008] It is the method of dividing that whose boundary used when carrying out division management of the space is a

hyperplane in K-D-B Tree [reference 1] and a rectangle field R-Tree It is SS-Tree about the method of dividing by [reference 2] and hypersphere. It is SR-Tree about the method of dividing space in the combination of [reference 3], a rectangle, and hypersphere. It is called [reference 4]. If N dimension vector space is furthermore rechanged into the space stretched by the characteristic vector about the covariance matrix of distribution of a prototype, more effective structuring is expectable to the computational complexity of retrieval [reference 5 and 6].

[0009] However, the application to the data with which the computational complexity and storage capacity for data-structure-izing increase exponentially to the increment in the number of dimension of a vector, and these methods are expressed by the high order former vector may be restricted as a matter of fact. Furthermore, to a fatal thing, the similarity based on an inner product is incalculable, if a test vector is not given. Therefore, when an inner product defines similarity, structuring of such data cannot be used.

[0010] [reference 1] -- JT. Robinson: "The K-D-B Tree: A Search Structure for Large Multidimensional Dynamic Indexes and" Proc. On ACM SIGMOD, pp.10-18 1981.[reference 2]A. Guttman: "R-trees: A dynamic index structure for spatial searching and" Proc. ACM SIGMOD, Boston USA pp.47-57, Jun.1984.[reference 3]DA. White and R.Jain: "Similarity indexing with the SS-tree," Proc. Of the 12th Int. Conf. On Data Engineering, New Orleans USA pp.323-331 Feb.. [1996] [reference 4] Katayama, Sato: "Proposal of the index structure for the maximum contiguity retrieval to SR-Tree:high dimension point data" ***** (D-I), vol.18-D-I no.8 pp.703-717, Aug. 1997. [reference 5] RF.Sproull: "Refinements to Nearest Neighbor Searching in k-Dimensional Trees and" Algorithmica 6 pp.579-589 1991.[reference 6] DA. Hite and R. Jain: "Similarity Indexing: Algorithms and Performance," Proc. On SPIE pp.62-73 1996. [0011] On the other hand, there are also the loose structuring and the "loose" algorithm somewhat "wise" which has attained mitigation of computational complexity by the search algorithm which does not use a statistical property. This kind of algorithm is the only strategy which can be taken to the similar retrieval based on an inner product. Among those, one of the most fundamental things is the algorithm of Friedman and others called a projection algorithm [reference 7]. [Reference 7] Friedman F. Baskett and L.J. Shustek: "An Algorithm for Finding Nearest Neighbors" IEEE Structuring of the data demanded as pretreatment with JH. Trans. On Computers, pp.1000-1006, and a Oct. 1975. projection algorithm is sorting for every component of a vector, and is structuring based on a phase. That is, when a prototype is a d-dimensional vector, the sorting list of d pieces will be generated.

[0012] Two kinds, the list V_j which stored j component value put in order by ascending order by this processing, and the list I_j which stored the corresponding prototype ID number, are acquired only for the number of dimension of a vector. That is, the $n+1$ st values V_j ($n+1$) are beyond the n-th values V_j (n) from the head of V_j . Moreover, j component value $Y_{I_j}(n)$ of the prototype $Y_{I_j}(n)$ which makes $I_j(n)$ an ID number, and (j) are in agreement with $V_j(n)$.

[0013] The principle of the projection algorithm for extracting 1 set of what is the closest to a test vector X from a prototype set is explained using drawing 13 . Retrieval is performed along with the sorting lists V_m and I_m of one piece chosen on suitable criteria. This supports having chosen m shaft among drawing. Since the data number sorting was carried out [the data number] by the component value is stored, as for I_m , the list on a list is reflecting the phase on m shaft correctly. The value nearest to the m component X of a test vector X (m) is first looked for from V_m . This is set to $V_m(j)$. The prototype corresponding to this is $Y_{I_m}(j)$. By a diagram, $Y_{I_m}(j)$ is Y1. Although Y1 is the closest to X about m component, the distance in the whole space is not necessarily the nearest.

[0014] Now, next, the distance rho of X and Y1 (X, Y1) is calculated. Then, only the prototype with which the value of m component belongs to an open interval (X (m) rho (X, Y1), X(m)+rho (X, Y1)) (the open interval A in drawing) may be closer to X than Y1, and it turns out that it is meaningful as an object of retrieval. In the example of drawing, the set of the prototype used as the candidate for retrieval is restricted further (X (m) rho (X, Y2), X(m)+rho (X, Y2)) (the open interval B in drawing) by inspecting the prototype Y2 near a degree about m component. Thus, it is a projection algorithm that mitigation of computational complexity is in drawing by making small the prototype set set as the object of retrieval based on the component value in 1-dimensional subspace.

[0015]

[Problem(s) to be Solved by the Invention] Friedman's and others projection algorithm mentioned above is a method for the similar retrieval based on distance, and was not able to be applied for the similar retrieval based on the absolute value of an inner product, or a square.

[0016]

[Means for Solving the Problem] In order to solve a technical problem mentioned above, according to this invention, from a prototype set described by multi-dimension vector To a data processor extracted as a retrieval result, a prototype of a predetermined individual similar to the 1st given test vector A pretreatment means to create a list which arranged an identifier and a component value of each prototype of said prototype set in order of a component value about each

component of said multi-dimension vector, and to calculate a square of a norm of each prototype, A test vector generation means to generate the 2nd test vector from said 1st test vector, A similarity count means to calculate similarity based on an absolute value of an inner product of the 1st test vector and each prototype concerned using a square of said list and a norm of each of said prototype, and the 1st and 2nd test vectors, By decision of abbreviation conditions based on a value of said similarity, it has an abbreviation control means which omits future processings, and an updating means to update a retrieval result based on a value of said similarity.

[0017] moreover, according to other modes of this invention, from a prototype set described by multi-dimension vector To a data-processing method extracted as a retrieval result, a prototype of a predetermined individual similar to the 1st given test vector A head end process which creates a list which arranged an identifier and a component value of each prototype of said prototype set in order of a component value about each component of said multi-dimension vector, and calculates a square of a norm of each prototype, A test vector generation production process which generates the 2nd test vector from said 1st test vector, A similarity count production process which calculates similarity based on an absolute value of an inner product of the 1st test vector and each prototype concerned using a square of said list and a norm of each of said prototype, and the 1st and 2nd test vectors, By decision of abbreviation conditions based on a value of said similarity, it has an abbreviation control production process of omitting future processings, and an updating production process which updates a retrieval result based on a value of said similarity.

[0018] By furthermore, the data-processing program which extracts a prototype of a predetermined individual which according to other modes of this invention gives and is similar to the 1st test vector of ***** from a prototype set described by multi-dimension vector as a retrieval result A head end process which creates a list which arranged an identifier and a component value of each prototype of said prototype set in order of a component value about each component of said multi-dimension vector, and calculates a square of a norm of each prototype, A test vector generation production process which generates the 2nd test vector from said 1st test vector, A similarity count production process which calculates similarity based on an absolute value of an inner product of the 1st test vector and each prototype concerned using a square of said list and a norm of each of said prototype, and the 1st and 2nd test vectors, A computer is made to perform an abbreviation control production process of omitting future processings, and an updating production process which updates a retrieval result based on a value of said similarity by decision of abbreviation conditions based on a value of said similarity.

[0019]

[Embodiment of the Invention] With the <operation gestalt of ** 1st> book operation gestalt, the relational expression of an inner product and distance was drawn and the projection algorithm was used as high-speed calculus of an inner product. Furthermore, in order to evaluate an absolute value, the inner product to the given test vector and the inner product to a test vector and the vector of hard flow were evaluated. Therefore, it is the hard structuring and the "hard" general-purpose thing which does not need the parameter which should be set up beforehand to a prototype set.

[0020] Before giving details of the algorithm of this operation gestalt, the definition of a problem setup, a phrase, and a mark is clarified.

[0021] :omega= {Y1, Y2, ..., YN} and Yj**Rd which set the set of the prototype Yj of N individual with which the problem to assume was expressed as a d-dimensional vector to omega -- at this time, k large prototypes are extracted from omega about the absolute value of measuring rhoG (X, Yj) to test vector X** Rd given suitably. However, measuring rhoG (X, Yj) is : [0022] defined as an inner product.

[External Character 1]

$$\rho_G(X, Y_j) = X^T G Y_j = \sum_{n=1}^{n=d} \sum_{m=1}^{m=d} G(m, n) X(m) Y_j(n) \quad (1)$$

Here, it wrote X (k) and Yj [a test vector X and k component value of Prototype Yj] (k), respectively.

[0023] the feature of this operation gestalt is the about the same as measuring rhoG (X, Yj) -- it is having adopted delta (Z, Yj) mentioned later as a function which gives a phase. It is equal to evaluating a prototype to descending of rhoG (X, Yj) and rhoG (- X, Yj) to extract a prototype to descending of the absolute value of rhoG (X, Yj). That is, if rhoG (- X, Yj) is large even if rhoG (X, Yj) is small, the absolute value of Yj of an inner product is large.

[0024] : which can divide rhoG (X, Yj) and rhoG (- X, Yj) into two steps of following processings -- Z=GXrhoG (X, Yj) =XTGY=(GX) TY=ZTY=rho (Z, Yj) rhoG(- X, Yj) =-XTGY=-(GX) TY=-ZTY=rho (- Z, Yj) -- rho (Z, Yj) and rho (- Z, Yj) are the inner products in orthonormal system here. On the other hand A ||Z+||Yj||2-2rho (Z, Yj) top type is transformed further. :||Z-Yj||2=(Z-Yj)T(Z-Yj)=||Z which develops the square of the distance of Z and Yj and obtains a degree type -- : which obtains a degree type -- 2rho(Z, Yj)-||Z||2 -- : which defines the right-hand side (or left part) of a =||Yj||2-||Z-Yj||2 top type as a new function delta (X, Yj) -- now [delta(- Z, Yj) =||Yj||2-||Z+Yj||2] like delta(Z, Yj)

$=\|Y_j\|^2 - \|Z - Y_j\|^2$ Suppose that the absolute value of k prototypes and an inner product with those test vectors X is given. The minimum value is set to x_i among the absolute values of these inner products. These k pieces presuppose that it is a recently contact in this time.

[0025] Since Prototype Y_j has the absolute value of an inner product smaller [the absolute value of an inner product with the newly given prototypes Y_j and X is smaller than x_i or] than all k prototypes if equal, or it is equal, it is guaranteed that it is not possible with k recently dots. That is, it can judge by whether a degree type is satisfied. : $(x_i \geq \delta(Z, Y_j)) \text{ ** } (x_i \geq \delta(-Z, Y_j))$ A dot set can be updated (2), thus recently. On the other hand, $\|Z - Y_j\|^2 \geq \|P(Z - Y_j)\|^2$ in which the following inequalities are materialized about the 2nd term of the right-hand side of the definition type of $\delta(Z, Y_j)$, however P are the orthogonal projection operators to subspace. Therefore, $\delta(Z, Y_j) \leq \|Y_j\|^2 - \|P(Z - Y_j)\|^2$ (3) Similarly $\delta(-Z, Y_j) \leq \|Y_j\|^2 - \|P(Z + Y_j)\|^2$ (4) A criteria (2) type is : $(x_i \geq \|Y_j\|^2 - \|P(Z - Y_j)\|^2) \text{ ** } (x_i \geq \|Y_j\|^2 - \|P(Z + Y_j)\|^2)$ rewritten as follows using (3) and (4) types. (5) for example If the difference about m component of a vector, square $\|Z(m) - Y_j(m)\|^2$ of the sum, and $\|Z(m) + Y_j(m)\|^2$ are calculated and the criteria of (5) types are fulfilled, it turns out that Y_j is not contained in a dot recently.

[0026] (5) Evaluation by the formula is effective in respect of computational complexity rather than it uses (2) types. If the computational complexity for evaluating (2) types is actually estimated, since it is calculable beforehand, it is not necessary to take into consideration the 1st term of the right-hand side. As for the 2nd term, the multiplication for a number of dimension of a vector and addition are needed. On the other hand, the multiplication for a number of dimension of subspace and addition are enough for (5) evaluation of expression. Therefore, the more the dimension of the subspace where (5) types are materialized is small, the more mitigation of computational complexity is expectable.

[0027] Hereafter, 1 operation gestalt of this invention is explained to details using a drawing.

[0028] [Operation gestalt 1] The configuration of the data retrieval equipment of this operation gestalt is shown in drawing 1. I/O device 101 is equipped with a means to input retrieval data and commands, such as a communication line and a keyboard, and a means to output retrieval results, such as a communications control means and a display. For example, in the case of the computer of a stand-alone, the data which this I/O device 101 consisted of the keyboard and the indicating equipment, transmitted the input data inputted from the keyboard to inner product count equipment 102, and was transmitted from inner product count equipment 102 is outputted to an indicating equipment.

[0029] On the other hand, in the case of the communication terminal connected to the communication line, this I/O device consists of a communication controller, and the input data inputted through the communication line is transmitted to similarity count equipment 102, and it transmits the data received from similarity count equipment 102 to the address specified through the communication line. The set of the prototype of N individual expressed by the d -dimensional vector is stored in the data base 103, and it is accessed by similarity count equipment 102.

[0030] The procedure of the <similarity count equipment 102>, next similarity count equipment 102 is explained using the flow chart of drawing 2.

[0031] First, it inspects whether there was any input from I/O device 101 at step S201, and if there is nothing, step S201 is repeated, and if it is, it will progress to step S202. At step S202, it inspects whether it is that to which input data updates the data of a data base, and if that is right, and that is not right, it will progress to step S203 to step S204. At step S203, pretreatment mentioned later is performed, and if it is completed, it will progress to step S201. At step S204, it inspects whether it is the thing of computation, and if this input data is so, and it is not so, it will progress to step S205 to step S201. At step S205, retrieval processing mentioned later is performed, and if it is completed, it will progress to step S201.

[0032] Pretreatment performed at step S203 using drawing 9 is explained.

[0033] With pretreatment, the sorting list about the square and each component value of a norm of the prototype belonging to a prototype set is created. Since the former only calculates and memorizes square $\|Y_j\|^2$ of the norm of Prototype Y_j , below, the latter processing is explained. A list of two kinds called the list V_j which stored j component value put in order by ascending order by this processing, and the list I_j which stored the corresponding prototype ID number is obtained only for the number of dimension of a vector.

[0034] n is set as 1 at step S801. At step S802, the pair of a n -component value and an ID number is created about each prototype of N individual. namely, -- : -- $\{(Y_1(n), n) \text{ -- } (-Y \text{ -- two -- } (-n \text{ -- } n \text{ --}) \dots) (Y_N(n), N) \text{ -- a step -- S -- 803 -- the above-mentioned pair -- the ascending order of a } n\text{-component value -- rearranging.}$

[0035] $\{(Y_{In}(1), (n), In(1)) \text{ -- } (-Y_{In}(2), (n), and In(2)) \text{ -- } V_n = \text{which considers the list of a component value, and the list of an ID number as Lists } V_n \text{ and } In \text{ separately ... } (Y_{In}(N), (n), In(N)) \text{ and below} \} \{Y_{In}(1), (n), Y_{In}(2) \text{ and } (n), \dots, Y_{In}(N) \text{ and } (n)\}$ (6) $In = \{In(1), In(2), \dots, In(N)\}$ At the (7) step S804, n is incremented, if it is over the dimension d of a vector, processing will be ended, otherwise, it progresses to step S802. The relation of the list of two kinds is as follows. The $n+1$ st values $V_j(n+1)$ are beyond the n -th values $V_j(n)$ from the head of V_j . Moreover, j

component value $Y_{lj}(n)$ of the prototype $Y_{lj}(n)$ which makes $l_j(n)$ an ID number, and (j) are in agreement with $V_j(n)$.

[0036] The retrieval processing performed at step S205 using drawing 3 is explained shortly. As an input of retrieval processing, the number k of the prototype demanded as the vector X for retrieval (it is called a test vector below), and metric tensor G and a retrieval result is given.

[0037] The index list of the base of vector space is created at the $:Z=GX$ step S302 which multiplies a test vector by metric tensor from the left at step S301, and acquires Vector Z . $:\lambda=$ which this is a list which defines the sequence of the base which applies the rejection conditions mentioned later, for example, is a list corresponding to descending of the absolute value of the component value of a test vector $X \{ \lambda_1, \lambda_2, \dots, \lambda_d \}$ (8) -- $:$ which writes L sets to be λ_L from the one smaller again -- $\lambda_L = \{ \lambda_{d-L+1}, \lambda_{d-L+2}, \dots, \lambda_d \}$ (9) -- m is further set as λ_1 .

[0038] At step S303, initialization processing of the variable $PTR(ed)$ and related is performed. This processing is explained using drawing 4.

[0039] The sorting list V_m of m component values is acquired at step S401. Step S402 is searched for the value nearest to m component value X of a test vector (m) from V_m , and the location is stored in $PTR+$ at it. namely, $:- |V_m(PTR+) - X(m)| \leq |V_m(j) - X(m)|$ and $**j** \{1, 2, \dots, N\}$ -- similarly, it searches for the value nearest to $-X(m)$ from V_m , and the location is stored in $PTR-$. namely, $:- |V_m(PTR-) + X(m)| \leq |V_m(j) + X(m)|$ and $**j** \{1, 2, \dots, N\}$ step S403 : which initializes a related variable as follows $PTR+L = PTR+1$, $BND+L = 0$, and $CAL+L = 0$ $PTR+H = PTR++k$ near sets are initialized at 1, $BND+H = 0$, $CAL+H = 0$ $PTR-L = PTR-1$, $BND-L = 0$, $CAL-L = 0$ $PTR-H = PTR-$, $BND-H = 0$, and the $CAL-H = 0$ $PTR = PTR+$ step S304. This processing is explained using drawing 5.

[0040] At step S501, set $N_0(X)$ will be initialized to empty class soon. At step S502, t is set as 1. At step S503, PTR later mentioned using drawing 6 is updated. Since the 1st term of the right-hand side of $:second = Im(PTR) \cdot \delta_s = 2 |rho(Z, Y_j)|$ which calculates the prototypes $Y_{Im}(PTR)$ and δ_s of ID number $Im(PTR)$ at step S504, however a top type is calculated with pretreatment, only read-out from storage is required.

[0041] step S505 -- near set N_t : which adds an ID number and the value of δ_s to 1 (X) -- $N_t(X) = N_{t-1}(X) + \{(s, \delta_s)\}$ -- if t is incremented at the} step S506 and k is exceeded, it will progress to step S507, otherwise, will return to step S503.

[0042] At step S507, ID number s corresponding to x_{it-1} and it for the minimum value of δ_s will be memorized as $taut-1$ in a set soon. Initialization of the near set to step S304 is completed above.

[0043] At step S305, t is set as $k+1$. Renewal of PTR is performed at step S306. This processing is explained using drawing 6 and 7.

[0044] At step S601, it is $BND-L = 0$ and $PTR-L$ inspect whether it is 0, and if that is right, and that is not right, it will progress to step S602 to step S603. : which performs the following processings at step S602 -- if it inspects, it comes out so and it is [whether 0 and $PTR-H$ have $BND-H$ equal to $PTR+H$, and] at $BND-L=1$ and the $Dx-L = \infty$ step S603 -- step S604 -- otherwise, it progresses to step S605. : which performs the following processings at step S604 -- if 0 and $PTR+L$ inspect how [smaller than $PTR-H$] it is, and $BND+L$ comes out so and there is at $BND-H = 1$ and the $Dx-H = \infty$ step S605 -- step S606 -- otherwise, it progresses to step S607. : which performs the following processings at step S606 -- if 0 and $PTR+H$ inspect whether they are two or more N , and $BND+H$ comes out so and there is at $BND-H=1$, $Dx-H = \infty$ $BND+L=1$, and the $Dx+L = \infty$ step S607 -- step S608 -- otherwise, it progresses to step S609. : which performs the following processings at step S608 -- at $BND+H=1$ and the $Dx+H = \infty$ step S609, the product of four numbers, $BND-L$, $BND-H$ and $BND+L$, and $BND+H$, inspects whether it is 1, if it comes out so and is, retrieval processing will be ended, otherwise, it progresses to step S610.

[0045] At step S610, $BND-L+CAL-L$ inspects whether it is 1, and if that is right, and that is not right, it will progress to step S611 to step S612. : which performs the following processings at step S611 -- if $BND-H+CAL-H$ inspects whether it is 1, comes out so and there is at $Dx-L = |V_m(PTR-L) - Z(m)|$ $CAL-L=1$ step S612 -- step S613 -- otherwise, it progresses to step S614. : which performs the following processings at step S613 -- if $BND+L+CAL+L$ inspects whether it is 1, comes out so and there is at $Dx-H = |V_m(PTR-H) - Z(m)|$ $CAL-H=1$ step S614 -- step S615 -- otherwise, it progresses to step S616. : which performs the following processings at step S615 -- if $BND+H+CAL+H$ inspects whether it is 1, comes out so and there is at $Dx+L = |V_m(PTR+L) - Z(m)|$ $CAL+L=1$ step S616 -- step S617 -- otherwise, it progresses to step S618. : which performs the following processings at step S617 -- if $Dx-L$ is smaller than any of $Dx-H$, $Dx+L$, and $Dx+H$ at $Dx+H = |V_m(PTR+H) - Z(m)|$ $CAL+H=1$ step S618 -- step S619 -- otherwise, it progresses to step S620.

[0046] At step S619, the following processings are performed, and in $:Dx=Dx-L$, $PTR=PTR-L$, and $CAL-L=0$ step S620 which end renewal of PTR of step S306, if $Dx-H$ is smaller than any of $Dx-L$, $Dx+L$, and $Dx+H$, and that is not

right, it will progress to step S621 to step S622.

[0047] At step S621, the following processings are performed, and in : $Dx=Dx-H$, $PTR=PTR-H$, and $CAL-H=0$ step S622 which end renewal of PTR of step S306, if $Dx+L$ is smaller than any of $Dx-L$, $Dx-H$, and $Dx+H$, and that is not right, it will progress to step S623 to step S624.

[0048] The following processings are performed at step S623, and at : $Dx=Dx+L$, $PTR=PTR+L$, and $CAL+L=0$ step S624 which end renewal of PTR of step S306, the following processings are performed, and by update process of : $Dx=Dx+H$ and $PTR=PTR+H$ which end renewal of PTR of step S306, and $CAL+H=0$ PTR, if the value of the variable relevant to PTR is changed and conditions are satisfied, the retrieval processing itself shown in drawing 3 will be ended.

[0049] By step S307, a degree type estimates (5) types by judgment processing later mentioned per drawing 8 at the : $\rho+=||Ys||-||Z||$ $\rho-=||Ys||-||Z||$ step S308 which calculates $\rho+$ and $\rho-$.

[0050] At step S309, the return value of the processing performed at step S308 judges whether it is FALSE, and if it is FALSE, and that is not right, it will progress to step S310 to step S312.

[0051] At step S310, the element corresponding to $xit-1$ will be deleted from a set soon. : which adds the prototype under current processing -- $Nt(X) < Nt-1(X) \{(\tau-1, xit-1)\} + \{(\text{Im}(PTR), Dx) \}$ -- at the step S311, the minimum value of Dx of the elements of $Nt(X)$ is stored in xit , the ID number is stored in τ , and it progresses to step S313.

[0052] At step S312, a degree type is performed, t is incremented at $t:=xi$ $t-1$ τ $\tau-1$ step S313 which progresses to step S313, if the result exceeds N , processing will be ended, otherwise, it progresses to step S306.

[0053] The function count performed by judgment processing of step S308 is explained using drawing 8.

[0054] At step S701, j is set as 1. A degree type is judged at $2Dx < -Dx-(Ys(\text{lambdaj})-X(\text{lambdaj}))^2$ step S703. : which performs the following processings at step S702 -- $\rho+=\rho+-(Ys(\text{lambdaj})-X(\text{lambdaj}))^2$ $\rho-=\rho-+(Ys(\text{lambdaj})+X(\text{lambdaj}))^2$ -- If satisfied, TRUE will be set as a return value and processing will be ended, otherwise, it progresses to step S704. At step S704, if j is incremented and the result exceeds the dimension d of a vector, FALSE will be set as a return value, and processing will be ended, otherwise, it will return to step S702.

[0055] $Nt(X)$ when ending at step S313 is outputted as a retrieval result.

[0056] The effect by the operation gestalt explained above is verified by computer simulation.

[0057] [computer simulation] -- the number of prototypes demanded as a retrieval result in order to verify the effectiveness of the operation gestalt mentioned above -- $k=10$ pieces and several prototypes -- computer simulation of retrieval was conducted to $N=10000$ pieces. dimension [of the : and the vector whose following items are experiment parameters]: -- $d=$ -- {-- : and CPU:PentiumIII whose item of the computer used for the 10 unit} experiment from 10 to 100 is as follows, and - (500MHz) main memory:128 MB-OS:Linux -2.0.36 -- in addition, programming language used C.

[0058] [Experiment procedure]

- (1) Generate the prototype set which consists of a d -dimensional vector N individual using a uniform random number.
- (2) Use a uniform random number and it is d dimensions. One metric tensor of vector space is generated.
- (3) Generate one d -dimensional test vector using a uniform random number.
- (4) Perform exhaustive search.
- (5) Perform retrieval by the proposal algorithm.

[0059] The average of the relative CPU time which shows the five above-mentioned procedures to a 10 times repeat and the following was calculated.

Relative CPU time = (CPU time of a proposal algorithm) $a /$ (CPU time of exhaustive search) experimental result is shown in drawing 10. The horizontal axis was shown for the dimension (number of dimensions) of a vector among drawing, and the axis of ordinate and the number N of prototypes were shown for the relative CPU time (CPU time ratio) as a parameter.

[0060] Drawing shows that it is not concerned with the number N of prototypes, but the relative CPU time is increasing to the primary order with the increment in the dimension of a vector. However, the inclination at the time of $N=10000$ pieces is very small. The value of the relative CPU time at this time indicates the sufficiently small value to be-dimensional [100] or 18% also by high order origin 2% by ten dimensions.

[0061] <Operation gestalt 2> The case where the similarity count mentioned above is applied to recognition equipment is explained.

[0062] Drawing 11 is drawing showing the configuration of the recognition equipment of this operation gestalt. The set of the prototype of N individual expressed by the d -dimensional vector is stored in the data base 103, and two or more prototypes exist to each category used as the candidate for recognition, respectively. Others are the same as that of drawing 1.

[0063] In for example, face recognition, the d-dimensional vector stored in a data base 103 can constitute by connecting the person's information with the image in two or more angles of an everybody object (Mr. A, Mr. B, ...), expression, etc. In such a case, by processing described below, an input image and k near images are chosen, the frequency of the person number connected with each can be totaled, and the highest person of frequency can be specified as he is a person corresponding to an image.

[0064] Moreover, the procedure of similarity count equipment 102 follows drawing 2 like the operation gestalt 1. However, in step S204-205, retrieval processing turns into recognition processing which used retrieval. This recognition processing is explained using the flow chart of drawing 12.

[0065] In drawing 12, step S301-313 are the same as that of drawing 3. At step S314, the frequency of the class number related with the ID number which will belong to a set soon is calculated. An ID number is rearranged into descending of the frequency of a class number at step S315. Thereby, a class candidate is obtained by order with a high possibility that a test vector belongs. Among these, a class with the largest frequency may be outputted as a recognition result, and a predetermined individual or all classes may be outputted as a recognition candidate in order of frequency.

[0066] In addition, even if it applies this invention to the equipment which consists of a single device, it may be applied to the system which consists of two or more devices. Moreover, the storage which memorized the program code of the software which realizes the function of the operation gestalt mentioned above may be supplied to equipment or a system, and you may attain by reading and performing the program code with which the computer in equipment or a system was stored in the storage.

[0067] Furthermore, by reading and performing the program code with which the computer in equipment or a system was stored in the storage, also when an above-mentioned function is realized, it is contained by processing of OS which it not only realizes directly the function of the operation gestalt mentioned above, but is working on a computer based on directions of the program code.

[0068] The storage which memorized the program code in these cases will constitute this invention.

[0069]

[Effect of the Invention] Since the similarity based on the absolute value of the inner product of a vector is calculable at a high speed according to this invention as explained above, based on this similarity, it is effective in the ability to perform retrieval of similar data and recognition of a class which belongs at a high speed to the given data by which vectorial representation was carried out.

[Translation done.]

(51) Int.Cl. ⁷	識別記号	F I	テーマコード(参考)
G 0 6 F 17/30	3 5 0	G 0 6 F 17/30	3 5 0 C 5 B 0 5 6
	2 1 0		2 1 0 D 5 B 0 7 5
17/16		17/16	K 5 L 0 9 6
G 0 6 T 7/00	3 0 0	G 0 6 T 7/00	3 0 0 C
			3 0 0 F

審査請求 未請求 請求項の数 7 O L (全 11 頁)

(21) 出願番号 特願2001-122755(P2001-122755)

(22) 出願日 平成13年4月20日 (2001. 4. 20)

(71) 出願人 000001007

キヤノン株式会社

東京都大田区下丸子3丁目30番2号

(72) 発明者 鷲澤 輝芳

東京都大田区下丸子3丁目30番2号キヤノン株式会社内

(74) 代理人 100090538

弁理士 西山 恵三 (外1名)

Fターム(参考) 5B056 BB42 HH00 HH03

5B075 NK06 NR12 PR06 QM08 QS20

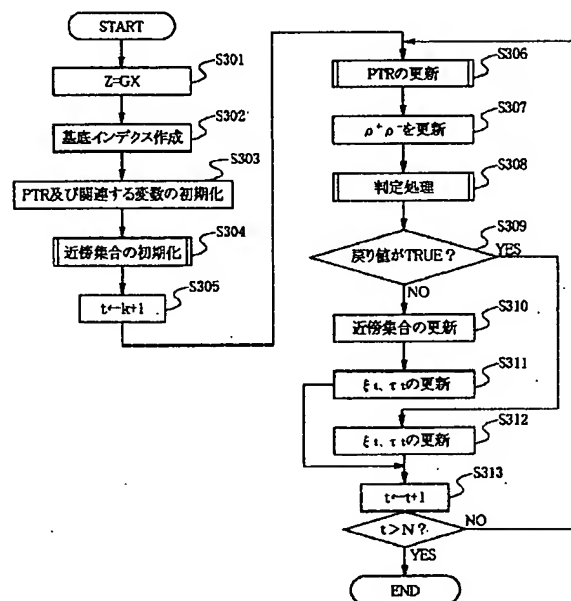
5L096 JA03 JA26

(54) 【発明の名称】 データ処理装置及びその方法、及びそのプログラム

(57) 【要約】

【課題】 与えられたベクトルデータに対して内積を基にした類似度によるデータ検索が高速に実行できるようにする。

【解決手段】 プロトタイプとなるベクトルの各成分に対するソーティング・リストを作成しておき、テストベクトル x 及び $-x$ の m 成分値に最も近い値をリストから探索し、それぞれの位置をPTR'及びPTRに格納する(S303)。 k 個の近傍集合の初期化を行い(S304)、PTRと関連する変数を更新する(S306)。順次プロトタイプ y_i を取り出し、判定処理によって、類似度が今までの最小値より大きければ、最小値に対応するプロトタイプと y_i を入れ替え、近傍集合及び最小値を更新する(S308～311)。



【特許請求の範囲】

【請求項1】 多次元ベクトルで記述されたプロトタイプ集合から、与えられた第1のテストベクトルに類似する所定個のプロトタイプを検索結果として抽出するデータ処理装置であって、

前記多次元ベクトルの各成分につき、前記プロトタイプ集合の各プロトタイプの識別子と成分値とを成分値の順に並べたリストを作成し、各プロトタイプのノルムの2乗を計算する前処理手段と、

前記第1のテストベクトルから第2のテストベクトルを生成するテストベクトル生成手段と、

前記リスト及び前記各プロトタイプのノルムの2乗と、第1及び第2のテストベクトルとを用いて、当該第1のテストベクトルと各プロトタイプとの内積の絶対値に基づく類似度を計算する類似度計算手段と、

前記類似度の値に基づく省略条件の判断により、以後の処理を省略する省略制御手段と、

前記類似度の値に基づいて検索結果を更新する更新手段とを有することを特徴とするデータ処理装置。

【請求項2】 上記テストベクトル生成手段が、前記第2のテストベクトルとして、前記第1のテストベクトルとノルムが等しく方向が逆であるようなベクトルを生成することを特徴とする請求項1に記載のデータ処理装置。

【請求項3】 前記プロトタイプ及び第1のテストベクトルが表現されているベクトル空間の次元を d とすると、前記類似度計算手段は、まず1次元部分空間に対して類似度を計算し、

前記省略制御手段が、当該1次元部分空間における類似度に基づく省略条件のを判断し、該省略条件を満足しなければ、部分空間の次元を上げて類似度計算を行うように前記類似度計算手段を制御し、

前記更新手段は、部分空間の次元が d に達してもなお省略条件を満足しないときに、検索結果を更新することを特徴とする請求項1に記載のデータ処理装置。

【請求項4】 前記類似度計算手段が、プロトタイプのノルムの2乗から第1のテストベクトルとプロトタイプの差の2乗を減じたものと、プロトタイプのノルムの2乗から第2のテストベクトルとプロトタイプの差の2乗を減じたものを用いて、類似度を計算することを特徴とする請求項1に記載のデータ処理装置。

【請求項5】 検索結果として抽出された前記所定個のプロトタイプが属するクラスに基づいて、前記第1のテストベクトルの属するクラスを認識する認識手段を有することを特徴とする請求項1に記載のデータ処理装置。

【請求項6】 多次元ベクトルで記述されたプロトタイプ集合から、与えられた第1のテストベクトルに類似する所定個のプロトタイプを検索結果として抽出するデータ処理方法であって、

前記多次元ベクトルの各成分につき、前記プロトタイプ

集合の各プロトタイプの識別子と成分値とを成分値の順に並べたリストを作成し、各プロトタイプのノルムの2乗を計算する前処理工程と、

前記第1のテストベクトルから第2のテストベクトルを生成するテストベクトル生成工程と、

前記リスト及び前記各プロトタイプのノルムの2乗と、第1及び第2のテストベクトルとを用いて、当該第1のテストベクトルと各プロトタイプとの内積の絶対値に基づく類似度を計算する類似度計算工程と、

前記類似度の値に基づく省略条件の判断により、以後の処理を省略する省略制御工程と、

前記類似度の値に基づいて検索結果を更新する更新工程とを有することを特徴とするデータ処理方法。

【請求項7】 多次元ベクトルで記述されたプロトタイプ集合から、与えられた第1のテストベクトルに類似する所定個のプロトタイプを検索結果として抽出するデータ処理プログラムであって、

前記多次元ベクトルの各成分につき、前記プロトタイプ集合の各プロトタイプの識別子と成分値とを成分値の順に並べたリストを作成し、各プロトタイプのノルムの2乗を計算する前処理工程と、

前記第1のテストベクトルから第2のテストベクトルを生成するテストベクトル生成工程と、

前記リスト及び前記各プロトタイプのノルムの2乗と、第1及び第2のテストベクトルとを用いて、当該第1のテストベクトルと各プロトタイプとの内積の絶対値に基づく類似度を計算する類似度計算工程と、

前記類似度の値に基づく省略条件の判断により、以後の処理を省略する省略制御工程と、

前記類似度の値に基づいて検索結果を更新する更新工程とをコンピュータに実行させるためのデータ処理プログラム。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、データの検索または認識を行なうデータ処理装置に関し、特にベクトルで表現されたデータ間の内積を基にした類似度によってデータを検索もしくは認識する装置に関するものである。

【0002】

【従来の技術】ベクトルで表現されたデータ間の類似度として内積が広く用いられている。例えば文字認識や音声認識システムでは、サンプリングされたデータを適当な基底で張られた特徴量空間に写像し、ベクトル表現されたデータをプロトタイプとして記憶しておく。新たに入力されたデータとプロトタイプとの内積を計算し、入力データを最も近いプロトタイプに対応するクラスに属するものとして同定する。最も効率の悪い計算方法は全数探索であり、その計算量はベクトルの次元とプロトタイプ数の積のオーダーになる。

【0003】内積の計算量が決定的な障害として認識さ

れるのは、データベース検索においてである。近年のコンピュータ処理能力の飛躍的発達によってデータベースは文書データのみならず、画像や音声といった非テキストデータをも蓄積できるようになった。これら非テキストデータを従来のキーワードで検索するためにはキーワードを付与しなければならない。キーワード付与の手間を避けたいなら特徴量ベクトルによる類似検索を行わなければならない。

【0004】また文書データを検索する場合でも、より柔軟な検索を可能にするために文書をベクトルで表現し、このベクトルに基づく類似検索を行うアルゴリズムも実現されている。このとき前記計算量が検索システム実現の本質的問題となる。通常のデータベースに格納されるデータ件数は数十万件を超える。従ってベクトルの次元数が1個増加する度に計算量が数十万増加するという絶望的な状況になる。このような状況を回避するためには、ベクトルの次元を減らすか、計算するデータ数を減らすしかない。

【0005】前者はデータの表現空間の次元を減らすことに等しいので、データ検索に必要な情報がベクトルの成分として十分表現されない可能性がある。後者は、検索結果として要求されているデータ数がデータの総数に比べて十分小さいときに意味のある方法論である。k-NN探索で取り扱う問題はこの種のものであり、いくつかの有効な方法が提案されている。

【0006】k-NN探索は、システムに記憶されているプロトタイプの集合から、テストベクトルに近いk個を探索し、それらのクラスを基に、テストベクトルのクラスを同定するための方法である。このような場合、テストベクトルに近いk個のプロトタイプを如何に高速に見つけることができるかが重要な課題の1つである。このような要求はデータベース検索においても存在する。検索ユーザーは、データベースに格納されている大量のデータのうち、自分が指定した検索キーに近いいくつかのデータだけが欲しいのであって、他のデータの、ましてや内積の値などに何の価値も見出さない。検索ユーザーのこのような要求に応えるための技術は、k-NN探索の高速アルゴリズムの目的と一致する。

【0007】プロトタイプの集合からテストベクトルに近いk個を抽出する場合、探索の計算量を軽減するために予めプロトタイプの集合を構造化しておくのが一般的である。構造化の際にデータの性質を反映させればさせるほど、探索の計算量の軽減が期待できる。例えばプロトタイプを階層的に構造化する場合では、プロトタイプが表現されているN次元ベクトル空間を分割するという操作を再帰的に繰り返すことによって達成される。

【0008】空間を分割管理するときに用いられる境界が超平面であるものをK-D-B Tree【文献1】、矩形領域で分割する方法をR-Tree【文献2】、超球で分割する方法をSS-Tree【文献3】、矩形と超球との組み合わせ

で空間を分割する方法をSR-Tree【文献4】という。更にN次元ベクトル空間をプロトタイプの分布の共分散行列に関する固有ベクトルで張られる空間に変換し直しておけば、探索の計算量に対して、より効果的な構造化が期待できる【文献5、6】。

【0009】しかしこれらの方法はデータ構造化のための計算量と記憶容量がベクトルの次元数の増加に対して指数関数的に増大してしまい、高次元ベクトルで表現されているデータへの応用が事実上制限されてしまう可能性がある。更に、致命的なことに、内積に基づく類似度は、テストベクトルが与えられなければ計算できない。従って、類似度を内積で定義した場合、このようなデータの構造化が利用できない。

【0010】【文献1】JT. Robinson: "The K-D-B Tree: A Search Structure for Large Multidimensional Dynamic Indexes," Proc. On ACM SIGMOD, pp.10-18, 1981.

【文献2】A. Guttman: "R-trees: A dynamic index structure for spatial searching," Proc. ACM SIGMOD, Boston, USA, pp.47-57, Jun.1984.

【文献3】DA. White and R. Jain: "Similarity indexing with the SS-tree," Proc. Of the 12th Int. Conf. On Data Engineering, New Orleans, USA, pp.323-331, Feb. 1996.

【文献4】片山, 佐藤: "SR-Tree: 高次元点データに対する最近接検索のためのインデックス構造の提案," 信学論(D-I), vol.18-D-I, no.8, pp.703-717, Aug. 1997.

【文献5】RF. Sproull: "Refinements to Nearest Neighbor Searching in k-Dimensional Trees," Algorithmica, 6, pp.579-589, 1991.

【文献6】DA. Hite and R. Jain: "Similarity Indexing: Algorithms and Performance," Proc. On SPIE, pp.62-73, 1996.

【0011】一方、統計的性質を利用しない"緩い"構造化と、少し"賢い"探索アルゴリズムによって計算量の軽減を達成しているアルゴリズムもある。この種のアルゴリズムが、内積に基づく類似検索に対して取り得る唯一の戦略である。そのうち最も基本的なもののひとつが射影アルゴリズムと呼ばれるFriedmanらのアルゴリズムである【文献7】。

【文献7】JH. Friedman, F. Baskett, and LJ. Shustek: "An Algorithm for Finding Nearest Neighbors," IEEE Trans. On Computers, pp.1000-1006, Oct. 1975.

射影アルゴリズムで前処理として要求されるデータの構造化は、ベクトルの各成分毎のソーティングであり、位相に基づく構造化である。つまり、プロトタイプがd次元ベクトルのときは、d個のソーティング・リストが生成されることになる。

【0012】この処理で、昇順に並べられたj成分値を格納したリスト V_i と、対応するプロトタイプID番号を格納したリスト I_i の2種類が、ベクトルの次元数だけ得られる。即ち、 V_i の先頭からn+1番目の値 $V_i(n+1)$ はn番目の値 $V_i(n)$ 以上である。また、 $I_i(n)$ をID番号とするプロトタイプ $Y_{i(i(n))}$ のj成分値 $Y_{i(i(n))}(j)$ が $V_i(n)$ と一致する。

【0013】プロトタイプ集合からテストベクトル X に最も近いもの1組を抽出するための射影アルゴリズムの原理を図13を用いて説明する。探索は、適当な基準で選択された1個のソーティング・リスト V_m と I_m に沿って行われる。これは、図中、m軸を選択したことに対応している。 I_m は成分値によってソーティングされたデータ番号が格納されているので、リスト上での並びがm軸上での位相を正確に反映している。まずテストベクトル X のm成分 $X(m)$ に最も近い値を V_m から探す。これを $V_m(i)$ とする。これに対応するプロトタイプは $Y_{m(i)}$ である。図では $Y_{m(i)}$ が Y_1 である。 Y_1 はm成分に関して X に最も近いが、全空間での距離が最も近いとは限らない。

【0014】さて次に、 X と Y_1 との距離 $\rho(X, Y_1)$ を計算する。すると、m成分の値が开区間 $(X(m) - \rho(X, Y_1), X(m) + \rho(X, Y_1))$ (図中の开区間A)に属するプロトタイプのみが、 Y_1 より X に近い可能性があり、探索の対象として意味があることがわかる。図の例では、m成分に関して次に近いプロトタイプ Y_2 を検査することによって、探索対象となるプロトタイプの集合が更に $(X(m) - \rho(X, Y_2), X(m) + \rho(X, Y_2))$ (図中の开区間B)に制限される。このように1次元部分空間での成分値をもとに、探索の対象となるプロトタイプ集合を小さくしていくことによって、計算量の軽減を図っているのが射影アルゴリズムである。

【0015】

【発明が解決しようとする課題】上述したFriedmanらの射影アルゴリズムは、距離に基づく類似検索のための方法であって、内積の絶対値、あるいは二乗に基づく類似検索のためには適用できなかった。

【0016】

【課題を解決するための手段】上述した課題を解決するために、本発明によれば、多次元ベクトルで記述されたプロトタイプ集合から、与えられた第1のテストベクトルに類似する所定個のプロトタイプを検索結果として抽出するデータ処理装置に、前記多次元ベクトルの各成分につき、前記プロトタイプ集合の各プロトタイプの識別子と成分値とを成分値の順に並べたリストを作成し、各プロトタイプのノルムの2乗を計算する前処理手段と、前記第1のテストベクトルから第2のテストベクトルを生成するテストベクトル生成手段と、前記リスト及び前記各プロトタイプのノルムの2乗と、第1及び第2のテストベクトルとを用いて、当該第1のテストベクトルと各プロトタイプとの内積の絶対値に基づく類似度を計算する類似度計算手段と、前記類似度の値に基づく省略条

件の判断により、以後の処理を省略する省略制御手段と、前記類似度の値に基づいて検索結果を更新する更新手段とを備える。

【0017】また、本発明の他の態様によれば、多次元ベクトルで記述されたプロトタイプ集合から、与えられた第1のテストベクトルに類似する所定個のプロトタイプを検索結果として抽出するデータ処理方法に、前記多次元ベクトルの各成分につき、前記プロトタイプ集合の各プロトタイプの識別子と成分値とを成分値の順に並べたリストを作成し、各プロトタイプのノルムの2乗を計算する前処理工程と、前記第1のテストベクトルから第2のテストベクトルを生成するテストベクトル生成工程と、前記リスト及び前記各プロトタイプのノルムの2乗と、第1及び第2のテストベクトルとを用いて、当該第1のテストベクトルと各プロトタイプとの内積の絶対値に基づく類似度を計算する類似度計算工程と、前記類似度の値に基づく省略条件の判断により、以後の処理を省略する省略制御工程と、前記類似度の値に基づいて検索結果を更新する更新工程とを備える。

【0018】更に、本発明の他の態様によれば、多次元ベクトルで記述されたプロトタイプ集合から、与えられた第1のテストベクトルに類似する所定個のプロトタイプを検索結果として抽出するデータ処理プログラムにより、前記多次元ベクトルの各成分につき、前記プロトタイプ集合の各プロトタイプの識別子と成分値とを成分値の順に並べたリストを作成し、各プロトタイプのノルムの2乗を計算する前処理工程と、前記第1のテストベクトルから第2のテストベクトルを生成するテストベクトル生成工程と、前記リスト及び前記各プロトタイプのノルムの2乗と、第1及び第2のテストベクトルとを用いて、当該第1のテストベクトルと各プロトタイプとの内積の絶対値に基づく類似度を計算する類似度計算工程と、前記類似度の値に基づく省略条件の判断により、以後の処理を省略する省略制御工程と、前記類似度の値に基づいて検索結果を更新する更新工程とをコンピュータに実行させる。

【0019】

【発明の実施の形態】<第1の実施形態>本実施形態では、内積と距離との関係式を導出し、内積の高速計算法として、射影アルゴリズムを利用した。更に、絶対値を評価するために、与えられたテストベクトルに対する内積と、テストベクトルと逆方向のベクトルに対する内積とを評価した。従って、プロトタイプ集合に対する“硬い”構造化や、予め設定すべきパラメタを必要としない汎用的なものである。

【0020】本実施形態のアルゴリズムの詳細を述べる前に、問題設定、語句と記号の定義を明らかにしておく。

【0021】想定する問題は、d次元ベクトルとして表現されたN個のプロトタイプ Y_i の集合を Ω とする：

$\Omega = \{Y_1, Y_2, \dots, Y_m\}, Y_i \in R^d$

このとき、適当に与えられたテストベクトル $X \in R^d$ に、計量 $\rho_c(X, Y_i)$ の絶対値に関して大きい k 個のプロトタイプを Ω から抽出する。ただし計量 $\rho_c(X, Y_i)$ は内積とし *

*で定義される：

[0022]

[外1]

$$\rho_c(X, Y_j) = X^T G Y_j = \sum_{n=1}^{n=d} \sum_{m=1}^{m=d} G(m, n) X(m) Y_j(n) \quad (1)$$

ここで、テストベクトル X 、プロトタイプ Y_i の k 成分値をそれぞれ $X(k)$ 、 $Y_i(k)$ と表記した。

【0023】本実施形態の特徴は、計量 $\rho_c(X, Y_j)$ と同じ位相を与える関数として、後述する $\delta(Z, Y_i)$ を採用した点である。 $\rho_c(X, Y_i)$ の絶対値の大きい順にプロトタイプを抽出するという事は、 $\rho_c(X, Y_i)$ と $\rho_c(-X, Y_i)$ の大きい順にプロトタイプを評価することに等しい。即ち、 $\rho_c(X, Y_i)$ が小さくても $\rho_c(-X, Y_i)$ が大きければ、 Y_i は内積の絶対値が大きい。

【0024】 $\rho_c(X, Y_i)$ 及び $\rho_c(-X, Y_i)$ は以下のような2段階の処理に分割できる：

$Z = GX$

$\rho_c(X, Y_i) = X^T G Y_i = (GX)^T Y_i = Z^T Y_i = \rho(Z, Y_i)$

$\rho_c(-X, Y_i) = -X^T G Y_i = -(GX)^T Y_i = -Z^T Y_i = \rho(-Z, Y_i)$

ここで $\rho(Z, Y_i)$ 及び $\rho(-Z, Y_i)$ は正規直交系における内積である。一方、 Z と Y_i との距離の2乗を展開して次式を得る：

$$\|Z - Y_i\|^2 = (Z - Y_i)^T (Z - Y_i) = \|Z\|^2 + \|Y_i\|^2 - 2\rho(Z, Y_i) \quad (2)$$

このようにして、最近傍点集合を更新することが出来る。一方、 $\delta(Z, Y_i)$ の定義式の右辺第2項に関して以下の不等式が成立する：

$$\|Z - Y_i\|^2 \geq \|P(Z - Y_i)\|^2$$

ただし、 P は部分空間への直交射影作用素である。従って

$$(\xi \geq \|Y_i\|^2 - \|P(Z - Y_i)\|^2) \wedge (\xi \geq \|Y_i\|^2 - \|P(Z + Y_i)\|^2) \quad (5)$$

例えば、ベクトルの m 成分に関する差と和の二乗 $\|Z(m) - Y_i(m)\|^2$ と $\|Z(m) + Y_i(m)\|^2$ を計算して、(5)式の判定条件を満たせば、 Y_i が最近傍点に含まれないことがわかる。

【0026】(5)式での評価は、(2)式を利用するよりも、計算量の面で効果的である。実際、(2)式を評価するための計算量を概算してみると、右辺第1項は、予め計算できるから、考慮する必要はない。第2項は、ベクトルの次元数分の乗算と加算が必要になる。一方(5)式の評価には、部分空間の次元数分の乗算と加算で十分である。よって、(5)式が成立する部分空間の次元が小さければ小さいほど、計算量の軽減が期待出来る。

【0027】以下、図面を用いて本発明の1実施形態を詳細に説明する。

【0028】〔実施形態1〕図1に本実施形態のデータ検索装置の構成を示す。入出力装置101は、通信回線やキーボード等の検索データやコマンドを入力する手段

※上式を更に変形して次式を得る：

$$2\rho(Z, Y_i) - \|Z\|^2 = \|Y_i\|^2 - \|Z - Y_i\|^2$$

10 上式の右辺(或いは左辺)を新たな関数 $\delta(X, Y_i)$ として定義する：

$$\delta(Z, Y_i) = \|Y_i\|^2 - \|Z - Y_i\|^2$$

同様に

$$\delta(-Z, Y_i) = \|Y_i\|^2 - \|Z + Y_i\|^2$$

今、 k 個のプロトタイプと、それらのテストベクトル X との内積の絶対値が与えられているとする。これら内積の絶対値のうち、最小値を ξ とする。この k 個が、現時点での最近接点であるとする。

【0025】新たに与えられたプロトタイプ Y_i と X との

20 内積の絶対値が ξ よりも小さいか等しければ、プロトタイプ Y_i は、 k 個の全てのプロトタイプよりも、内積の絶対値が小さいか等しいので、 k 個の最近傍点で有り得ないことが保証される。即ち、次式を満足するかどうかで判定できる：

$$\star \delta(Z, Y_i) \leq \|Y_i\|^2 - \|P(Z - Y_i)\|^2 \quad (3)$$

同様に

$$\delta(-Z, Y_i) \leq \|Y_i\|^2 - \|P(Z + Y_i)\|^2 \quad (4)$$

30 判定条件(2)式は、(3)及び(4)式を用いて以下のように書き換えられる：

★

$$(\xi \geq \|Y_i\|^2 - \|P(Z - Y_i)\|^2) \wedge (\xi \geq \|Y_i\|^2 - \|P(Z + Y_i)\|^2) \quad (5)$$

と、通信制御手段や表示装置等の検索結果を出力する手段を備える。例えばスタンドアロンのコンピュータの場合は、該入出力装置101はキーボードと表示装置から成り、キーボードから入力された入力データを内積計算装置102へ送信し、内積計算装置102から送信されたデータを表示装置に出力する。

40 【0029】一方、通信回線に接続されている通信端末の場合は、該入出力装置は通信制御装置から成り、通信回線を介して入力された入力データを類似度計算装置102に送信し、類似度計算装置102から受信したデータを通信回線を介して指定されたアドレスに送信する。データベース103には、 d 次元ベクトルで表現された N 個のプロトタイプの集合が格納されており、類似度計算装置102によってアクセスされる。

【0030】＜類似度計算装置102＞次に、類似度計算装置102の処理手順を図2のフローチャートを用いて説明する。

50

【0031】まず、ステップS201で入出力装置101から入力があったかどうかを検査し、なければステップS201を繰り返し、あればステップS202へ進む。ステップS202では、入力データがデータベースのデータを更新するものかどうかを検査し、そうであればステップS203へ、そうでなければステップS204へ進む。ステップS203では、後述する前処理を実行し、それが終了したらステップS201へ進む。ステップS204では該入力データが計算処理のものかどうかを検査し、そうであればステップS205へ、そうでなければステップS201へ進む。ステップS205では、後述する検索処理を実行し、それが終了したらステップS201へ進む。

【0032】図9を用いてステップS203で実行される前処理について説明する。

【0033】前処理では、プロトタイプ集合に属するプロトタイプのノルムの2乗と、各成分値に関するソーティング・リストを作成する。前者はプロトタイプ Y_i のノルムの2乗 $\|Y_i\|^2$ を計算して記憶しておくだけで、以下では後者の処理を説明する。この処理で、昇順に並べられた j 成分値を格納したリスト V_i と、対応するプロトタイプID番号を格納したリスト I_i という2種類のリストが、ベクトルの次元数だけ得られる。

【0034】ステップS801で n を1に設定する。ステップS802で、 N 個のプロトタイプそれぞれについて、 n 成分値とID番号のペアを作成する。即ち：

$$\{(Y_1(n), n), (Y_2(n), n), \dots, (Y_N(n), N)\}$$

ステップS803で上記ペアを n 成分値の昇順に並べ替える。

$$\{(YIn(1)(n), In(1)), (YIn(2)(n), In(2)), \dots, (YIn(N)(n), In(N))\}$$

以下、成分値の並びとID番号の並びを別々にリスト V_n 、 In とする：

$$V_n = \{Y_{In(1)}(n), Y_{In(2)}(n), \dots, Y_{In(N)}(n)\} \quad (6)$$

$$I_n = \{I_n(1), I_n(2), \dots, I_n(N)\} \quad (7)$$

ステップS804では n をインCREMENTし、ベクトルの次元 d を超えていたら処理を終了し、そうでなければステップS802へ進む。2種類のリストの関係は次のとおりである。 V_i の先頭から $n+1$ 番目の値 $V_i(n+1)$ は n 番目の値 $V_i(n)$ 以上である。また、 $I_i(n)$ をID番号とするプロトタイプ $Y_{I_i(n)}$ の i 成分値 $Y_{I_i(n)}(i)$ が $V_i(n)$ と一致する。

【0036】今度は図3を用いてステップS205で実行される検索処理を説明する。検索処理の入力として、検索のためのベクトル X （以下テストベクトルと呼ぶ）と、計量テンソル G 、検索結果として要求されているプロトタイプの個数 k が与えられている。

【0037】ステップS301ではテストベクトルに計量テンソルを左から乗じてベクトル Z を得る：

$$Z = GX$$

ステップS302ではベクトル空間の基底のインデクスリストを作成する。これは後述する棄却条件を適用する基底

の順序を定めるリストであり、例えば、テストベクトル X の成分値の絶対値の大きい順に対応するリストである：

$$\Lambda = \{\lambda_1, \lambda_2, \dots, \lambda_d\} \quad (8)$$

また小さいほうから L 個の集合を Λ_L と書く：

$$\Lambda_L = \{\lambda_{d-L+1}, \lambda_{d-L+2}, \dots, \lambda_d\} \quad (9)$$

更に m を λ_1 に設定する。

【0038】ステップS303ではPTR及び関連する変数の初期化処理を実行する。この処理を図4を用いて説明する。

【0039】ステップS401で m 成分値のソーティング・リスト V_m を取得する。ステップS402ではテストベクトルの m 成分値 $X(m)$ に最も近い値を V_m から探索し、その位置をPTR'に格納する。即ち：

$$|V_m(PTR')X(m)| \leq |V_m(j)X(m)|, \forall j \in \{1, 2, \dots, N\}$$

同様に、 $-X(m)$ に最も近い値を V_m から探索し、その位置をPTR'に格納する。即ち：

$$|V_m(PTR') + X(m)| \leq |V_m(j) + X(m)|, \forall j \in \{1, 2, \dots, N\}$$

ステップS403では、関連する変数は以下のように初期化する：

$$PTR'_L = PTR'_1, BND'_L = 0, CAL'_L = 0$$

$$PTR'_H = PTR'_1, BND'_H = 0, CAL'_H = 0$$

$$PTR'_L = PTR'_1, BND'_L = 0, CAL'_L = 0$$

$$PTR'_H = PTR'_1, BND'_H = 0, CAL'_H = 0$$

$$PTR = PTR'$$

ステップS304では k 個の近傍集合の初期化を行う。この処理を図5を用いて説明する。

【0040】ステップS501では近傍集合 $N_k(X)$ を空集合に初期化する。ステップS502では t を1に設定する。ステップS503では、図6を用いて後述するPTRの更新を行う。ステップS504ではID番号 $I_m(PTR)$ のプロトタイプ $Y_{I_m(PTR)}$ と Δ_m を計算する：

$$S = I_m(PTR)$$

$$\Delta_m = 2|\rho(Z, Y_i)|$$

ただし上式の右辺第1項は前処理で計算されているので、記憶装置からの読み出しだけで良い。

【0041】ステップS505では近傍集合 $N_{k-1}(X)$ にID番号と Δ_m の値とを追加する：

$$N_k(X) = N_{k-1}(X) + \{(S, \Delta_m)\}$$

ステップS506で t をインCREMENTし、 k を超えればステップS507へ進み、そうでなければステップS503へ戻る。

【0042】ステップS507では近傍集合のなかで Δ_m の最小値を ϵ_{t-1} 、またそれに対応するID番号 s を τ_{t-1} として記憶する。以上でステップS304の近傍集合の初期化を終了する。

【0043】ステップS305では t を $k+1$ に設定する。ステップS306ではPTRの更新を実行する。この処理を図6、7を用いて説明する。

【0044】ステップS601では BND'_L が0かつ PTR'_L が0かどうかを検査し、そうであればステップS602へ、そ

うでなければステップS603へ進む。ステップS602では以下の処理を実行する：

$$BND'_L = 1, Dx'_L = \infty$$

ステップS603では BND'_H が0かつ PTR'_H が PTR'_L と等しいかどうかを検査し、そうであればステップS604へ、そうでなければステップS605へ進む。ステップS604では以下の処理を実行する：

$$BND'_H = 1, Dx'_H = \infty$$

ステップS605では BND'_L が0かつ PTR'_L が PTR'_H より小さいかどうかを検査し、そうであればステップS606へ、そうでなければステップS607へ進む。ステップS606では以下の処理を実行する：

$$BND'_H = 1, Dx'_H = \infty$$

$$BND'_L = 1, Dx'_L = \infty$$

ステップS607では BND'_H が0かつ PTR'_H がN2以上かどうかを検査し、そうであればステップS608へ、そうでなければステップS609へ進む。ステップS608では以下の処理を実行する：

$$BND'_H = 1, Dx'_H = \infty$$

ステップS609では BND'_L と BND'_H 及び BND'_L と BND'_H の4個の数の積が1かどうかを検査し、そうであれば検索処理を終了し、そうでなければステップS610へ進む。

【0045】ステップS610では $BND'_L + CAL'_L$ が1かどうかを検査し、そうであればステップS611へ、そうでなければステップS612へ進む。ステップS611では以下の処理を実行する：

$$Dx'_L = |V_n(PTR'_L) - Z(m)|$$

$$CAL'_L = 1$$

ステップS612では $BND'_H + CAL'_H$ が1かどうかを検査し、そうであればステップS613へ、そうでなければステップS614へ進む。ステップS613では以下の処理を実行する：

$$Dx'_H = |V_n(PTR'_H) - Z(m)|$$

$$CAL'_H = 1$$

ステップS614では $BND'_L + CAL'_L$ が1かどうかを検査し、そうであればステップS615へ、そうでなければステップS616へ進む。ステップS615では以下の処理を実行する：

$$Dx'_L = 2 V_n(PTR'_L) - Z(m)$$

$$CAL'_L = 1$$

ステップS616では $BND'_H + CAL'_H$ が1かどうかを検査し、そうであればステップS617へ、そうでなければステップS618へ進む。ステップS617では以下の処理を実行する：

$$Dx'_H = |V_n(PTR'_H) - Z(m)|$$

$$CAL'_H = 1$$

ステップS618では Dx'_L が Dx'_H 、 Dx'_L 、 Dx'_H のいずれよりも小さければステップS619へ、そうでなければステップS620へ進む。

【0046】ステップS619では以下の処理を実行し、ステップS306のPTRの更新を終了する：

$$Dx = Dx'_L, PTR = PTR'_L, CAL'_L = 0$$

ステップS620では Dx'_H が Dx'_L 、 Dx'_L 、 Dx'_H のいずれより

も小さければステップS621へ、そうでなければステップS622へ進む。

【0047】ステップS621では以下の処理を実行し、ステップS306のPTRの更新を終了する：

$$Dx = Dx'_H, PTR = PTR'_H, CAL'_H = 0$$

ステップS622では Dx'_L が Dx'_L 、 Dx'_H 、 Dx'_H のいずれよりも小さければステップS623へ、そうでなければステップS624へ進む。

【0048】ステップS623では以下の処理を実行し、ステップS306のPTRの更新を終了する：

$$Dx = Dx'_L, PTR = PTR'_L, CAL'_L = 0$$

ステップS624では以下の処理を実行し、ステップS306のPTRの更新を終了する：

$$Dx = Dx'_H, PTR = PTR'_H, CAL'_H = 0$$

PTRの更新処理では、PTRに関連する変数の値を変更し、条件が満足されれば、図3に示した検索処理そのものを終了する。

【0049】ステップS307では、次式で ρ^+ 及び ρ^- を計算する：

$$\rho^+ = \|Y_t\| - \|Z\|$$

$$\rho^- = \|Y_t\| - \|Z\|$$

ステップS308では、図8につき後述する判定処理によって(5)式を評価する。

【0050】ステップS309では、ステップS308で行った処理の戻り値がFALSEかどうかを判定し、FALSEならステップS310へ、そうでなければステップS312へ進む。

【0051】ステップS310では、近傍集合から ξ_{t-1} に対応する要素を削除し、現在処理中のプロトタイプを追加する：

$$N_t(X) \leftarrow N_{t-1}(X) \setminus \{(\tau_{t-1}, \xi_{t-1})\} \cup \{(I_n(PTR), Dx)\}$$

ステップS311では、 $N_t(X)$ の要素のうちの Dx の最小値を ξ_t に、そのID番号を τ_t に格納し、ステップS313へ進む。

【0052】ステップS312では次式を実行し、ステップS313へ進む：

$$\xi_t = \xi_{t-1}$$

$$\tau_t = \tau_{t-1}$$

ステップS313では t をインCREMENTし、その結果が N を超えたら処理を終了し、そうでなければステップS306へ進む。

【0053】ステップS308の判定処理で実行する関数計算を図8を用いて説明する。

【0054】ステップS701では j を1に設定する。ステップS702では以下の処理を実行する：

$$\rho^+ = \rho^+ - (Y_s(\lambda_j) - X(\lambda_j))^2$$

$$\rho^- = \rho^- - (Y_s(\lambda_j) + X(\lambda_j))^2$$

$$Dx \leftarrow Dx - (Y_s(\lambda_j) - X(\lambda_j))^2$$

ステップS703では次式を判定し、満足したら戻り値にTRUEを設定して処理を終了し、そうでなければステップS704へ進む。ステップS704では、 j をインCREMENTし、

その結果がベクトルの次元dを超えたら、戻り値にFALSEを設定して処理を終了し、そうでなければステップS702へ戻る。

【0055】ステップS313で終了したときの、 $N_k(X)$ が検索結果として出力される。

【0056】以上説明した実施形態による効果を計算機実験によって検証する。

【0057】〔計算機実験〕上述した実施形態の有効性を検証するために、検索結果として要求されるプロトタイプ数 $k=10$ 個、プロトタイプ数 $N=10000$ 個に対して探索の計算機実験を行った。以下の項目が実験パラメタである：

・ベクトルの次元： $d=\{10から100まで、10きざみ\}$

実験に使用した計算機の諸元は以下のとおりである：

・CPU:PentiumIII,(500MHz)

・メインメモリ:128MB

・OS:Linux-2.0.36

尚、プログラム言語はCを用いた。

【0058】〔実験手順〕

(1) 一様乱数を用いて、 d 次元ベクトル N 個より成るプロトタイプ集合を生成する。

(2) 一様乱数を用いて、 d 次元ベクトル空間の計量テンソルを1個、生成する。

(3) 一様乱数を用いて、 d 次元のテストベクトルを1個、生成する。

(4) 全数探索を行う。

(5) 提案アルゴリズムによる探索を行う。

【0059】上記5つの手順を10回繰り返し、以下に示す相対CPU時間の平均値を計算した。

相対CPU時間=(提案アルゴリズムのCPU時間)/(全数探索のCPU時間)

実験結果を図10に示す。図中、ベクトルの次元(number of dimensions)を横軸、相対CPU時間(CPU time ratio)を縦軸、プロトタイプ数 N をパラメタとして示した。

【0060】図よりプロトタイプ数 N に関わらず、ベクトルの次元の増加に伴って相対CPU時間が1次のオーダーで増加していることがわかる。しかし $N=10000$ 個のときの傾きは非常に小さい。このときの相対CPU時間の値は、10次元で2%、100次元でも18%と、高次元でも十分小さい値を示している。

【0061】＜実施形態2＞上述した類似度計算を認識装置に適用する場合について説明する。

【0062】図11は、本実施形態の認識装置の構成を示す図である。データベース103には、 d 次元ベクトルで表現された N 個のプロトタイプの集合が格納されており、認識対象となる各カテゴリに対してそれぞれ複数のプロトタイプが存在する。その他は図1と同様である。

【0063】データベース103に格納される d 次元ベ

クトルは、例えば顔認識の場合は、各人物(Aさん、Bさん、...)の複数の角度、表情等における画像に、その人物の情報を関係付けることによって、構成することが出来る。このような場合、下記に述べる処理によって、入力画像と近い k 個の画像が選択され、それぞれに関係付けられた人物番号の頻度を集計し、頻度の最も高い人物を、画像に対応する人物であると特定することが出来る。

【0064】また、類似度計算装置102の処理手順は、実施形態1と同様に図2に従う。但し、ステップS204~205では、検索処理は、検索を用いた認識処理となる。この認識処理について図12のフローチャートを用いて説明する。

【0065】図12において、ステップS301~313は、図3と同一である。ステップS314では、近傍集合に属するID番号に関連付けられたクラス番号の頻度を計算する。ステップS315では、クラス番号の頻度の大きい順に、ID番号を並べ替える。これにより、テストベクトルが属する可能性が高い順にクラス候補が得られる。このうち最も頻度の大きいクラスを認識結果として出力してもよいし、頻度順に所定個あるいは全部のクラスを認識候補として出力してもよい。

【0066】尚、本発明は、単一の機器からなる装置に適用しても、複数の機器から構成されるシステムに適用してもよい。また、上述した実施形態の機能を実現するソフトウェアのプログラムコードを記憶した記憶媒体を、装置あるいはシステムに供給し、装置あるいはシステム内のコンピュータが記憶媒体に格納されたプログラムコードを読み出して実行することによって達成してもよい。

【0067】更に、装置あるいはシステム内のコンピュータが記憶媒体に格納されたプログラムコードを読み出して実行することによって、上述した実施形態の機能を直接実現するばかりでなく、そのプログラムコードの指示に基づいて、コンピュータ上で稼動しているOSなどの処理により、上述の機能を実現される場合も含まれる。

【0068】これらの場合、そのプログラムコードを記憶した記憶媒体は本発明を構成することになる。

【0069】

【発明の効果】以上説明したように、本発明によれば、ベクトルの内積の絶対値に基づく類似度を、高速に計算することができるので、この類似度に基づいて、与えられたベクトル表現されたデータに対して、類似データの検索や属するクラスの認識を高速に実行することができるという効果がある。

【図面の簡単な説明】

【図1】本発明に係る一実施形態のデータ検索装置の構成を示すブロック図である。

【図2】距離計算処理の手順を示すフローチャートであ

る。

【図3】検索処理の詳細手順を示すフローチャートである。

【図4】PTRの初期化手順を示すフローチャートである。

る。

【図5】近傍集合の初期化手順を示すフローチャートである。

ある。

【図6】PTRの更新手順を示すフローチャートである。 *

* 【図7】PTRの更新手順を示すフローチャートである。

【図8】判定処理の手順を示すフローチャートである。

【図9】前処理の手順を示すフローチャートである。

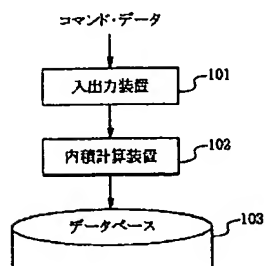
【図10】計算機実験の結果を示す図である。

【図11】認識装置の構成を示すブロック図である。

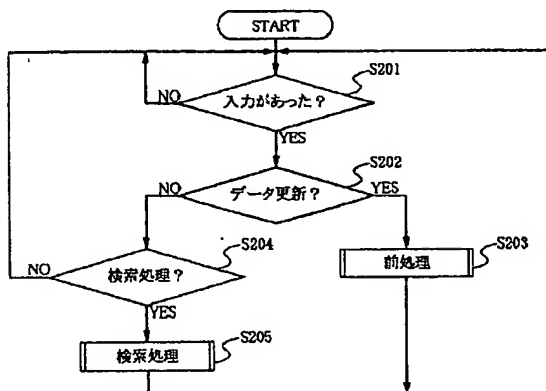
【図12】認識処理の詳細手順を示すフローチャートである。

【図13】射影アルゴリズムを説明する図である。

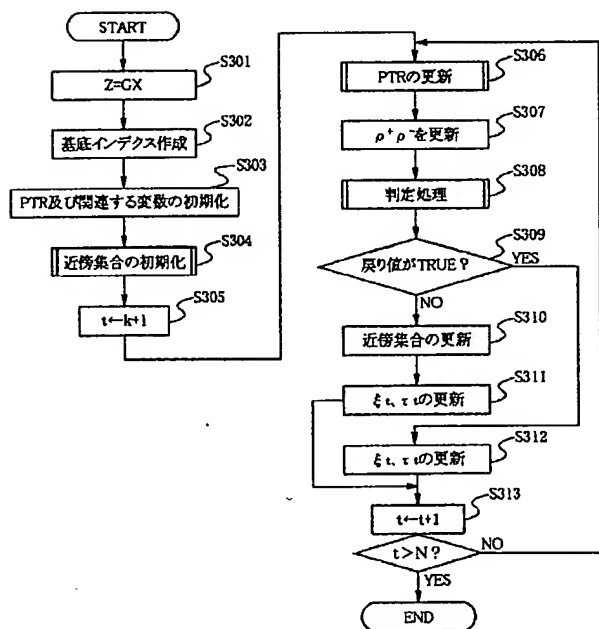
【図1】



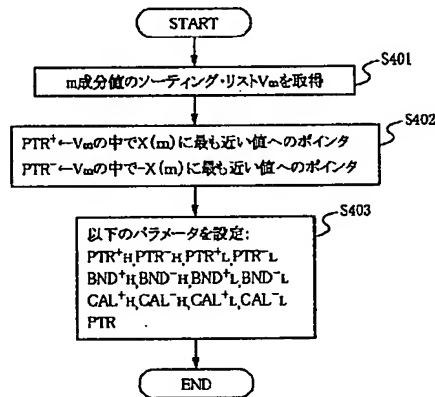
【図2】



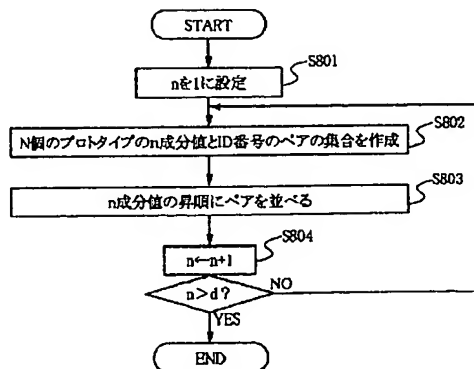
【図3】



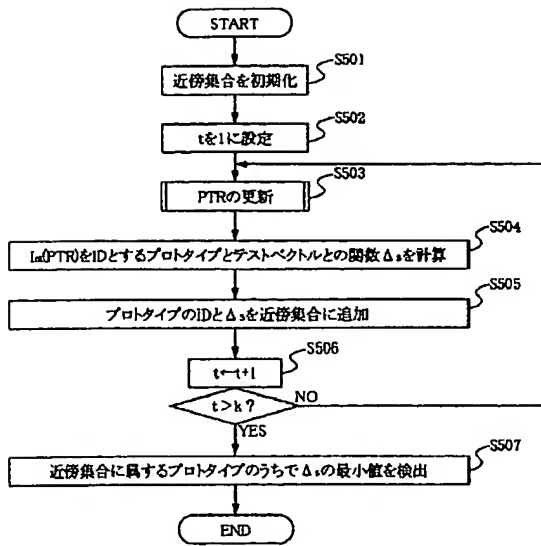
【図4】



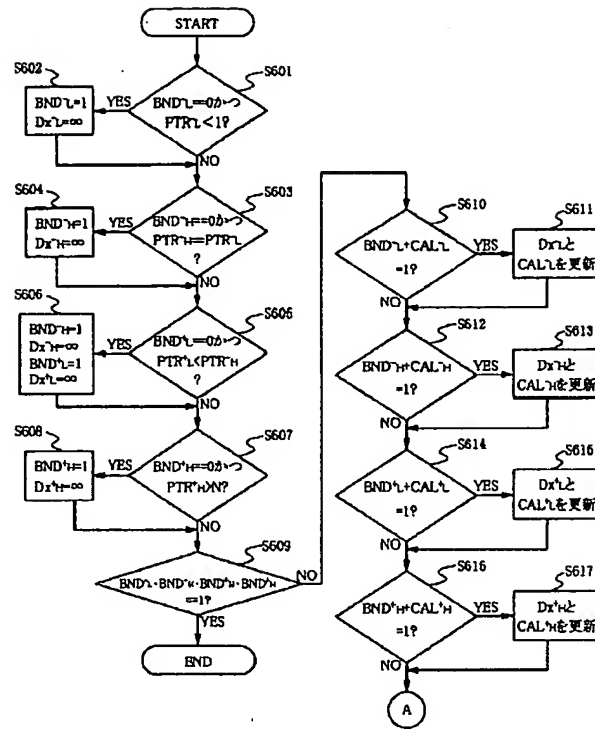
【図9】



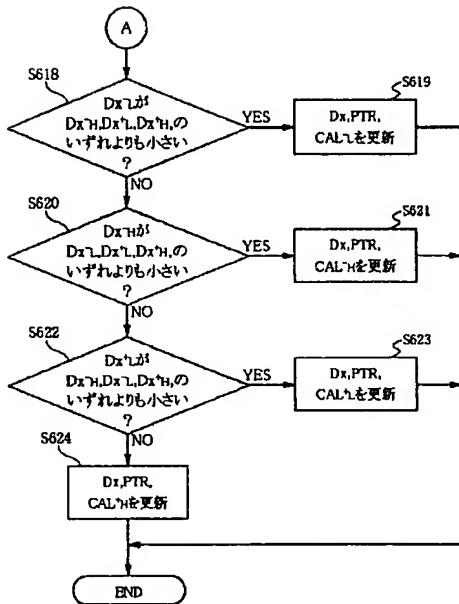
【図5】



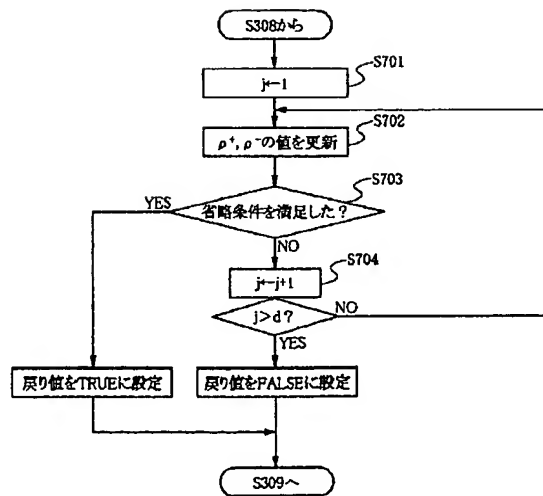
【図6】



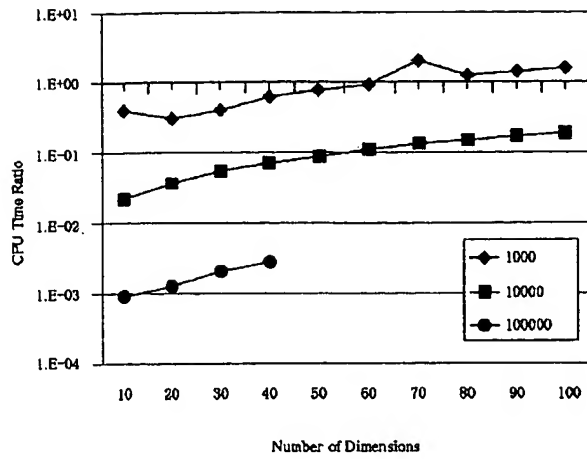
【図7】



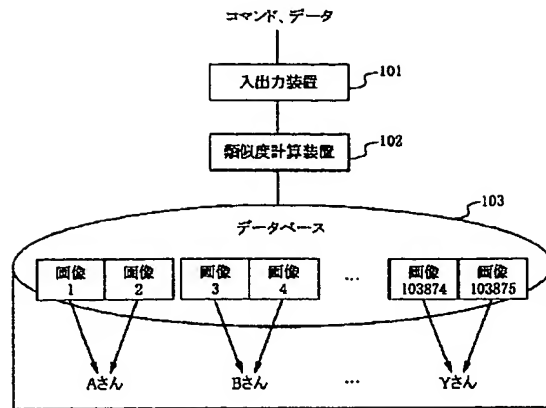
【図8】



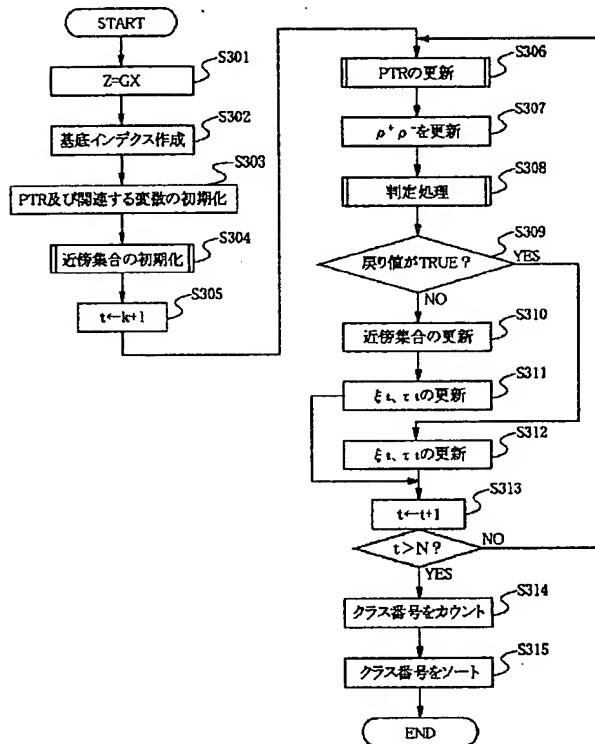
【図10】



【図11】



【図12】



【図13】

